# An Integral Security Kernel[1]

RAFAEL ALVAREZ, LEANDRO TORTOSA, JOSE FRANCISCO VICENT, and ANTONIO ZAMORA
Departamento de Ciencia de la Computación e Inteligencia Artificial
Universidad de Alicante
Campus de Sant Vicente, Ap. Correos 99, E-03080, Alicante
SPAIN

[1] *This work was partially supported by Generalitat Valenciana grant number GV04B-462.*

*Abstract:* - As the user base of the Internet has grown tremendously, the need for secure services has increased accordingly. Most secure protocols, in digital business and other fields, use a combination of symmetric and asymmetric cryptography, random generators and hash functions in order to achieve confidentiality, integrity, and authentication. Our proposal is an integral security kernel based on a powerful mathematical scheme from which all of these cryptographic facilities can be derived. The kernel requires very little resources and has the flexibility of being able to trade off speed, memory or security; therefore, it can be efficiently implemented in a wide spectrum of platforms and applications, either software, hardware or low cost devices. Additionally, the primitives are comparable in security and speed to well known standards.

*Key-Words:* - cryptography, security, integration, public-key, private-key, hash, random-generator.

## 1 Introduction

The user base of the Internet has increased amazingly during the latest years, and the connection speeds have seen dramatic improvements in most parts of the world. This has transformed the Web from a research network into a perfect marketplace, where businesses can set up facilities for what is generally known as ecommerce. The emergence of mobile and wireless computing has made new applications possible, allowing customers to take advantage of these services using small and convenient devices such as cell phones or PDAs.

Digital business is one of the most appealing possibilities of the internet. Usually, the web is used to show the products, much like in a traditional catalog, and to take the orders from the customers, as in a conventional shop. The most common way to pay is the usage of credit cards, though other methods are possible. Nevertheless, if we want this way of commerce to be successful, security must be a primary concern.

Despite the new and exciting possibilities, the Internet is still vulnerable to malicious attacks. The feeling of insecurity, that many have associated with digital business services, can only be fought with mathematically proven algorithms. Because of this, businesses have demanded protocols and services which are more secure and bring confidence to the end user.

Most protocols in digital business employ symmetric cryptography to transfer large quantities of data, while asymmetric cryptography is used to swap session keys, digital signatures, etc. Additionally, hash functions can be used in order to improve efficiency and data integrity.

Our proposal integrates these three basic components, obtaining a security kernel which can be the basis of many protocols. Since cryptographic algorithms are extremely diverse in nature, scope and requirements, this integration is highly beneficial since it allows for cheap mass production, and ease of design of new secure systems which could use the kernel as a black box.

Our cryptographic kernel is based on the powers of a block upper triangular matrix, which is a very flexible technique. It can be adjusted to satisfy memory and speed requirements and be implemented successfully either in hardware or software. Another advantage is that the same basic mathematical scheme can be used to build private key cryptosystems, public key cryptosystems and hash functions. Therefore, we only require to implement this technique once in order to provide these three types of algorithms, integrating a full cryptographic kernel in a single low cost device. This is a remarkable new concept, that shows how useful this technique can be in cryptography.

Symmetric ciphers can either be block or stream ciphers. The difference is that stream ciphers process input data bit by bit and block ciphers take a group of bits from the input and process them like a whole; being the former more efficient in general. In our

security kernel we employ a stream cipher as the symmetric component.

Most stream ciphers are based on the Vernam cipher scheme, in which a keystream generator outputs a random sequence, that is then XORed with the cleartext to obtain the ciphertext, and XORed again on the destination to retrieve the original cleartext. In practice, this generator is a pseudorandom generator [12,18,19]: a deterministic algorithm, with a reproducible output completely determined by its input. A pseudorandom generator takes an input value called seed and generates an output sequence, which is not really random, it just appears to be so for practical purposes. This means that to reproduce the sequence on the destination, we only need to transfer the seed which is, eventually, the key of the cipher.

RC4 (see [14,16]) is probably one of the most popular stream ciphers nowadays. It is a commercial algorithm and was developed by RSA Data Security, Inc. The algorithm is very simple and fast, using just some addition, modulus and substitution operations working with bytes. RC4 is secure because it can be in an enormous amount of states (about $2^{1700}$). RC4 is used in many secure systems such as SSL [8].

Regarding the security of the keystream, the Blum Blum Shub [2] (BBS) algorithm is a well known pseudorandom generator considered to be very secure. The BBS generator is secure due to the intractableness of factoring big numbers.

One of the greatest problems with symmetric cryptography is key distribution, because both parties must share a secret key before any communication takes place. This turns out to be very difficult when the communication is being done over an insecure channel (such as the Internet). Another inconvenience, is that you cannot perform digital signatures, so the receiver cannot be sure who has sent the message.

A public key cryptosystem allows the communication between both parties without the need of sharing any secret key, also providing means for digital signature. Each party has a pair of keys: a private key and a public key. Introduced by Diffie and Hellman [6] in the mid 70's, public key cryptography has been widely used in insecure communication networks.

Many asymmetric algorithms have been proposed, being the most popular RSA [17], because of its simplicity. It has survived numerous attacks but it requires a key of considerable length.

Asymmetric cryptosystems employ, generally, much longer keys than symmetric ones. They are also, commonly, much slower since they are based on complex mathematical calculations requiring more computing power. In practice, they are used to cipher the session key (secret) of each individual message or transaction.

The rest of the paper is divided as follows: in section 2 we describe the kernel and its facilities in detail; in section 3 we present some applications in which this kernel could be useful; finally, we present some conclusions in section 4.

## 2 Description

Our kernel is based on the powers of a block upper triangular matrix (BUTM) defined over $Z_p$, with $p$ prime. As we take the different powers of a BUTM, we have as a result a sequence of matrices of very long period that has great properties in terms of randomness. Each element of the sequence (each BUTM) can be processed to obtain a series of values that conform an output sequence with great statistical values. This scheme is simple enough to be really fast but incorporates enough complexity to present great cryptographic properties.

Consider the block upper triangular matrix $M$ defined as

$$M = \begin{bmatrix} A & X \\ O & B \end{bmatrix}, \tag{1}$$

whose entries lie in $Z_p$, where $A$ is an $r \times r$ matrix, $B$ is an $s \times s$ matrix and $X$ is an $r \times s$ matrix.

The following result establishes the expression of the different powers of the matrix $M$. It also defines matrix $X^{(h)}$ in terms of $A$, $B$ and $X$.

*Theorem 1* Let $M$ be the block upper triangular matrix given by (1).
Taking $h$ as a non negative integer then

$$M^h = \begin{bmatrix} A^h & X^{(h)} \\ O & B^h \end{bmatrix}, \tag{2}$$

where

$$X^{(h)} = \begin{cases} 0 & \text{if } h = 0, \\ \sum_{i=1}^{h} A^{h-i} X B^{i-1} & \text{if } h \geq 1. \end{cases} \tag{3}$$

Also, if $0 \leq t \leq h$ then

$$X^{(h)} = A^t X^{(h-t)} + X^{(t)} B^{h-t}. \tag{4}$$

It can also be proven that

$$X^{(a+b)} = A^a X^{(b)} + X^{(a)} B^b .$$

We fix matrices $A$ and $B$ and choose randomly matrix $X$, next we apply expression (4) to obtain the following succession of matrices:

$$X^{(2)}, X^{(3)}, X^{(4)}, \ldots \quad (5)$$

One of the basic properties that this sequence must hold is that its period must be very long (at least a period of $2^{128}$).

Now, we analyze the way we can get long periods for the sequence given by (5).

The key for the solution of this problem can be found in [10,15]: let

$$f(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{r-1} x^{r-1} + x^r$$

and

$$g(x) = b_0 + b_1 x + b_2 x^2 + \cdots + b_{s-1} x^{s-1} + x^s$$

be two primitive polynomial in $Z_{p[x]}$ and let $\overline{A}$ and $\overline{B}$ be the corresponding companion matrices;
let $P$ and $Q$ be two nonsingular matrices such that $A = P\overline{A}P^{-1}$ and $B = Q\overline{B}Q^{-1}$. With this construction, the order of matrix $M$ of (1) is

$$lcm(p^r - 1,\ p^s - 1). \quad (6)$$

The value of $p$ or the sizes of $A$ and $B$ do not need to be very large in order to achieve long periods, as shown in table 1, which shows some results for the period in terms of the parameters $p$, $r$ and $s$.

Table 1. Order of M for different values of p, r and s

| p | r | s | Digits | p | r | s | Digits |
|---|---|---|---|---|---|---|---|
| 3 | 32 | 31 | 30 | 19 | 16 | 19 | 39 |
| | 48 | 47 | 39 | | 32 | 31 | 57 |
| | 64 | 63 | 47 | | 64 | 63 | 98 |
| 5 | 32 | 31 | 38 | 31 | 16 | 15 | 40 |
| | 30 | 33 | 39 | | 32 | 31 | 64 |
| | 64 | 63 | 61 | | 64 | 63 | 111 |
| 7 | 24 | 27 | 39 | 251 | 12 | 13 | 46 |
| | 32 | 31 | 43 | | 32 | 31 | 76 |
| | 64 | 63 | 70 | | 64 | 63 | 168 |
| 11 | 22 | 21 | 39 | 257 | 9 | 10 | 40 |
| | 32 | 31 | 50 | | 32 | 31 | 93 |
| | 64 | 63 | 67 | | 64 | 63 | 169 |

The value appearing in the column *digits* represents the number of decimal digits of the period of the binary sequence (the integer $2^{128}$ has 39 decimal digits). Observe that the values taken for $r$ and $s$ are relatively prime, in order to optimize the period.

## 2.1 The Symmetric Component

To cipher large amounts of information efficiently, we need a private key cryptosystem. For that purpose, we can build a stream cipher using the mathematical base of the kernel by taking advantage of its great randomness properties. We first create a good pseudorandom generator and, once we have that, we use it as the keystream generator in a Vernam cipher scheme, taking the seed of the generator as the key of the stream cipher. This pseudorandom generator can also be used to generate general purpose random numbers such as session keys, challenge values, etc.

We apply expression (4) to obtain the sequence of matrices (5). For each matrix $X^{(h)}$, for $h = 2, 3, \ldots$, we establish a bit extraction operation which can be as simple as adding all the elements of $X^{(h)}$ obtaining a new element $x^{(h)}$, for $h = 2, 3, \ldots$, in $Z_p$ from which we take the least significant bit, $b^{(h)}$, of its binary expression; or as complex as required and taking as many bits per iteration as needed. In this way, we have the sequence of bits

$$b^2, b^3, b^4, \ldots \quad (7)$$

This sequence is then filtered by the following process, improving security and bias:

$$c^{(i)} = b^{(i)} \oplus c^{(i-1)},\ i = 2, 3, 4, \ldots; c^{(1)} = 0. \quad (8)$$

Once we have a proper keystream, ciphering the plaintext is as simple as XORing the keystream with it bit by bit. To decipher we XOR the keystream again with the ciphertext and retrieve the original plaintext. The seed of the generator is shared by both parties so that they can reproduce correctly the keystream.

The algorithm has been compared with the BBS pseudorandom generator (see [1]), achieving comparable results in terms of the randomness of the keystream and being a lot faster (in the order of $10^3$ times). A comparison with the RC4 stream cipher has also yielded comparable results in randomness and similar speed in software. Further optimizations are being studied and could make the algorithm even faster.

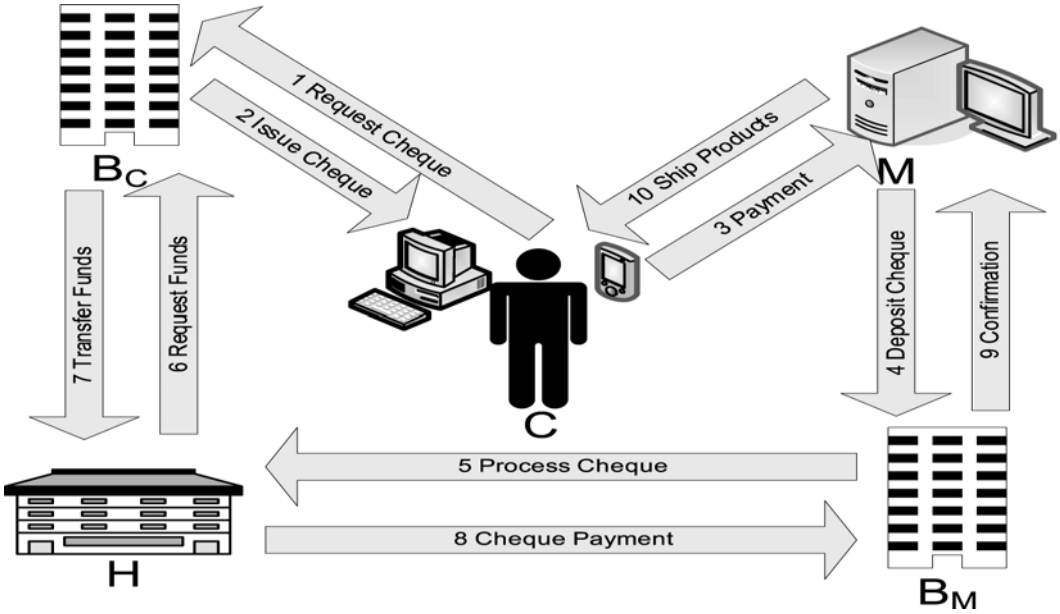


Fig.1. Electronic cheque payment system

### 2.2 The Asymmetric Component

Defining the operator $\otimes$ as

$$X^{(a)} \otimes X^{(b)} = X^{(a+b)}, \qquad (9)$$

set $G = \{X^{(0)}, X^{(1)}, X^{(2)}, X^{(3)}, \ldots\}$ has a finite group structure and its order can be taken as large as needed to make our scheme secure.

The key exchange scheme between two users $U$ and $V$, proposed for our kernel, is:

1. $U$ and $V$ accord values for $p$, $n$, $A$, $B$ and $X$
2. $U$ generates a random number $k$ and computes $A^k$, $B^k$ and $X^{(k)}$
3. $V$ generates a random number $m$ and computes $A^m$, $B^m$ and $X^{(m)}$
4. The numbers $k$ and $m$ are respectively the private keys of $U$ and $V$
5. The pairs $(X^{(k)}, B^k)$ and $(X^{(m)}, B^m)$ are respectively the public keys of $U$ and $V$
6. $U$ computes $X^{(k+m)} = A^k X^{(m)} + X^{(k)} B^m$
7. $V$ computes $X^{(m+k)} = A^m X^{(k)} + X^{(m)} B^k$

With this scheme users $U$ and $V$ share matrix $X^{(k+m)}$ in $G$.

The computation of $A^k$, $A^m$, $B^k$, $B^m$, $X^{(k)}$ and $X^{(m)}$ can be done efficiently adapting the existing quick exponentiation algorithm in $Z_p$ (see [9]).

It is computationally infeasible, for an attacker, to know the shared key $X^{(m+k)}$ without the previous knowledge of $k$ and $m$, because the problem the attacker would be facing is in the order of complexity of the discrete logarithm problem (see [13]).

The scheme described previously can be adapted to perform digital signature using a similar technique to the ElGamal [7] cryptosystem. Since the kernel requires little resources, it is suitable for low power or low cost environments.

### 2.3 The Hash Component

Taking the mathematical base of the kernel we can also build a hash function. We can use the pseudorandom generator as a diffusion and compression mechanism accumulating its results over a fixed length register. The stream cipher proposed can be also adapted to perform a hash function in the way shown in [18].

## 3 Applications

Our integral security kernel can be used by any digital business protocol requiring security at any level, like A/V content distribution systems [4], anonymous peer to peer systems [5], certified email systems [11], online payment systems [3], etc. It can be implemented on any platform (PC, dedicated hardware, PDA, latest generation of cell phones, smart cards) and data transport system (Internet, wireless networks, satellites, terrestrial digital transmissions, etc.), being capable of adapting to the technological evolutions in the communications sector.

It is efficient and easy to implement either in

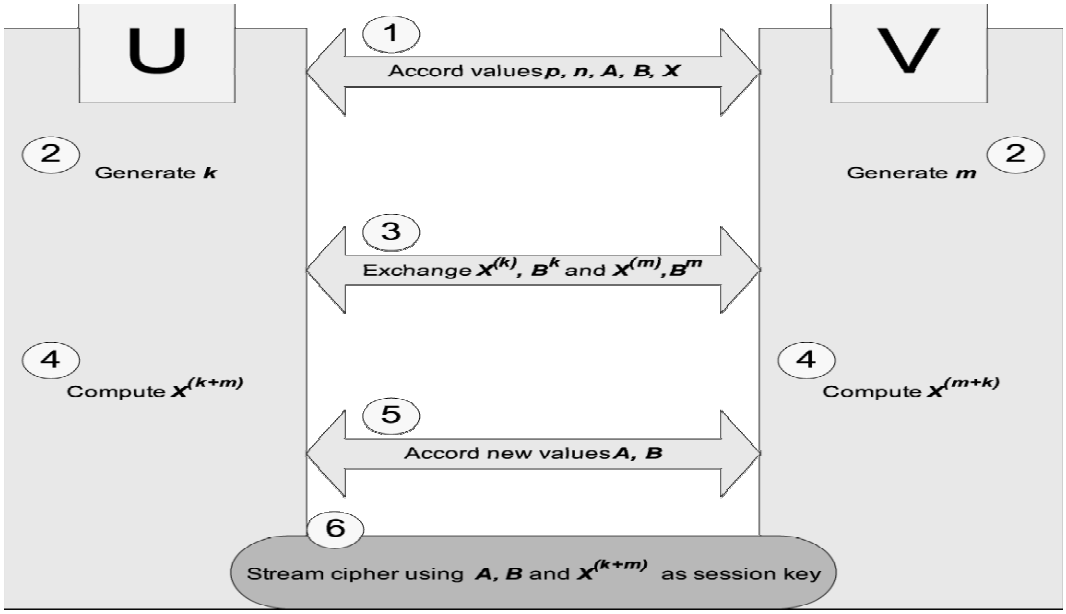

Fig.2. Secure communication scheme

hardware or software and requires very little resources, making possible its implementation in a wide spectrum of devices, specially those of low cost. In this way, confidentiality (ciphered information), integrity (no alteration warranty) and authentication (identity verification using digital signature) are assured in the communications.

As an application example of the kernel, we can take an electronic cheque payment system involving five parties: the client ($C$) and his bank ($B_C$), the merchant ($M$) and his bank ($B_M$), and a clearing house ($H$) that reconciles bank transfers and processes cheques.

- The client purchases some goods and sends the corresponding electronic cheque to the merchant.
- The merchant sends the check to his bank to validate and deposit it.
- The merchant's bank sends the check to the clearing house in order to receive payment.
- The clearing house requests the required funds from the client's bank. Then, both banks update the corresponding accounts.
- Once the electronic cheque has been validated and correctly processed, the merchant proceeds to send the goods to the client.

For each of the communication channels established between the different parties (see figure 1), we need to guarantee confidentiality, authentication, and information integrity. For that purpose we require the usage of symmetric and asymmetric cryptography, a random generator and a hash function (operations offered by the proposed kernel).

In this way, the kernel provides all the means for a secure communication between two parties, as shown in figure 2:

1. First, both parties must establish values for $p$, $n$, $A$, $B$ and $X$.
2. Then, $U$ generates a random $k$ of sufficient length, $V$ generates $m$ in the same way.
3. $U$ sends $V$ values $X^{(k)}$ and $B^k$, $V$ does the same sending $U$ the values $X^{(m)}$ and $B^m$.
4. $U$ computes $X^{(k+m)}$ and $V$ computes $X^{(m+k)}$, since both parties reach the same result they now share this secret key.
5. $U$ and $V$ can agree on new values for $A$ and $B$.
6. Taking the new $A$ and $B$, along with $X^{(k+m)}$, we have the session key for our secure channel, using the kernel's stream cipher.

## 4 Conclusion

In this paper we have proposed an integral security kernel that provides confidentiality, integrity and authentication in any digital business protocol requiring security at any level.

It is based on the powers of a block upper triangular matrix, a very flexible technique capable of being adjusted to satisfy memory and speed requirements and implemented successfully either in hardware or software.

From this basic technique we can derive pseudorandom generators, symmetric cryptosystems, asymmetric cryptosystems and hash functions; therefore, requiring a single implementation to

provide these key types of algorithms. This integration is highly beneficial since it allows for cheap mass production and ease of design of new secure products.

The security of the kernel is directly comparable to well known standards such as BBS (session key generation), RC4 (symmetric cryptography) and RSA (asymmetric cryptography), with similar or better performance in many cases.

*References:*


[1] Alvarez, R., Climent, J. J., Tortosa, L., Zamora, A., A Pseudorandom Bit Generator Based on Block Upper Triangular Matrices, *LNCS Web Engineering*, Vol. 2722, 2003, pp. 299-300

[2] Blum, L., Blum, M., Shub, M., A Simple Unpredictable Pseudorandom Number Generator, *SIAM J. Comput.*, Vol. 15, 1986, pp. 364-383

[3] Chaum, D., On-line Cash Checks, *Proc. Eurocrypt'89 LNCS*, Vol. 434, 1990, pp. 288-293

[4] Conrado, C., Kamperman, F., Schrijen, G., Jonker, W., Privacy in an Identity-based DRM System. *Proc. Int. Workshop on Database and Expert Systems Applications (DEXA'03)*, 2003, pp. 389-395

[5] Datta, A. K., Gradinariu, M., Raynal, M., Simon, G., Anonymous Publish-Subscribe in P2P Networks, *Proc. Int. Parallel and Distributed Processing Symposium (IPDPS'03)*, 2003, pp. 74-82

[6] Diffie, W., Hellman, M., New directions In Cryptography, *IEEE Trans. Information Theory*, Vol. 22, 1976, pp. 644-654

[7] Elgamal, T., A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms, *IEEE Trans. Inform. Theory*, Vol. 31, 1985, pp. 469-472

[8] Freier, A. O., Karlton, P., Kocher, P. C., The SSL Protocol Version 3.0., *Internet Draft Netscape*, 1996

[9] Gordon, D. M., A Survey of Fast Exponentiation Methods, *Journal of Algorithms*, Vol. 27, 1998, pp. 129-146

[10] Hoffman, K., Kunze, R., Linear Algebra, *Prentice-Hall*, New Jersey, 1971

[11] Imamoto, K., Sakurai, K., Certified E-mail Systems Using Public Notice Board, *Proc. Int. Workshop on Database and Expert Systems Applications (DEXA'03)*, 2003, pp. 460-466

[12] Menezes, A., van Oorschot, P., Vanstone, S., Handbook of Applied Cryptography, *CRC Press*, Florida, 2001

[13] Menezes, A., Wu, Y-H., The Discrete Logarithm Problem in GL(n,q), *Ars Combinatoria*, Vol. 47, 1997, pp. 22-32

[14] Mister, S., Tavares, S., Cryptanalysis of RC4-like Ciphers, *Select Areas In Cryptography LNCS*, Vol. 1556, 1998, pp. 131-143

[15] Odoni, R. W. K., Varadharajan, V., Sanders, P. W., Public Key Distribution in Matrix Rings, *Electronic Letters*, Vol. 20, 1984, 386-387

[16] Rivest, R., The RC4 Encryption Algorithm, *RSA Data Security Inc*, 1992

[17] Rivest, R., Shamir, A., Adleman, L., A Method for Obtaining Digital Signatures and Public Key Cryptosystems, *ACM Communications*, Vol. 21, 1978, pp. 120-126

[18] Schneier, B., Applied Cryptography Second Edition: protocols, algorithms and source code in C, *John Wiley and Sons*, New York, 1996

[19] Stallings, W., Cryptography and Network Security: Principles and Practice. Third Edition, *Prentice Hall*, New Jersey, 2003